Deep Reinforcement Learning Guided by a Library of Possibly Unreliable Advice

Nizam Makdoud, Jérôme Kodjabachian, Marc Schoenauer

Apprentissage par Reinforcement profond guidé par ensemble de politiques sources

Nizam Makdoud[1, 2], Jérôme Kodjabachian[1], et Marc Schoenauer[2]

[1]Thalès ThereSIS, France
[2]Inria Tau, Université Paris-Saclay, France

3 juin 2020

Résumé

Les capacités d'apprentissage impressionnantes des humains sont dues, dans une large mesure, à leur capacité à réutiliser les informations provenant de diverses sources. Transférer la compétence d'un agent constitue donc l'un des moyens les plus efficaces pour initialiser un agent sur une nouvelle tache. Cependant, sans garanties, l'imitation aveugle de conseils peut être préjudiciable. La raison réside dans l'incapacité d'un agent à évaluer correctement la valeur de ces conseils. Pour tirer des enseignements de conseils éventuellement peu fiables, nous proposons d'intégrer la connaissance d'une bibliothèque de politiques de conseillers (dite sources), en utilisant comme proxy la fonction de valeur. Cette fonction d'évaluation permet de quantifier la valeur d'une action (et donc d'un conseil). En outre, elle permet non seulement de choisir les meilleurs conseils parmi un ensemble proposé mais aussi d'entraîner un agent (dit target) à surpasser la compétence des politiques sources. Notre approche ne nécessite les conseillers qu'en phase d'entraînement et est robuste aux conseils inadéquats. Nous démontrons empiriquement sur un ensemble de tâches standard la résilience aux conseils peu fiables de notre algorithme et sa capacité à produire un transfert de compétences qui surpasse l'ensemble des politiques sources.

Mots-clef : Apprentissage par Renforcement Profond, Transfert Learning.

1 Introduction

Reinforcement Learning is a promising paradigm for decision-making under uncertainty. It captures several core characteristics of human learning abilities and demonstrates several successes in a large variety of challenging tasks, e.g., in Robotics [ABC+18] and Games [MK13]. Despite its successes, sample inefficiency [K+03] prevents its application in environments where interaction is costly. One promising solution to enhance the sample efficiency of Reinforcement Learning algorithms lies in the reuse of knowledge acquired from previously explored environments or tasks.

Indeed, a learning agent should reuse the knowledge acquired from other tasks. However, depending on the similarity between previous tasks and the current one, advice given could be misleading. Because of the difficulty to estimate in general the proximity between tasks, Transfer Learning algorithms should be resilient to poor advice. Unreliable advice may easily occur : For instance, hand-crafted reward functions are commonly used to guide an agent. Assuming an agent navigates in a maze to reach a goal, its distance to the goal could be reused as a guide via the proxy of an additional reward (that rewards the agent for being close to the goal). Unsurprisingly, this reward shaping would fail by leading the agent to deadly sub-optimal solutions depending on the architecture of the maze. It could then be tempting not to use any guidance scheme. Nonetheless, without guidance and, in this case, adequate exploration, reaching the goal, may be nearly impossible. Formal and empirical reasons [NHR99] explain this failure, but generally, hand-crafted guidance schemes are susceptible to be abused by a Reinforcement Learning algorithm.The complexity of the design of a robust guidance scheme leads us to tackle the problem of robust transfer learning to automatically reuse a library of policies without risking catastrophic negative transfer. In this work, we propose to reuse the knowledge from previously learned policies to automatically assess the value of each advisor in order to select the adequate advice (or none of them !) from a pool of potentially unreliable ones.

Prior works (see Section 5 for a detailed discussion) reuse

advice to modify the student's policy directly. Among those, *actor-driven guidance* tries to guide an agent by direct modification of its policy. Residual learning approaches [SATK18, JB$^+$19] propose to learn a corrective policy applied to a base policy provided by an expert. Inspired by the Imitation Learning literature [NMA18, VHSea17], an alternative guidance scheme is to imitate the advice that seems the most promising. However, if we hypothesize that, during the early epochs, the identification of valuable advice is merely random, direct imitation by modifying the student's policy may lead to catastrophic updates by forcing the student to imitate tragic advice.

In this work, we propose *LEarning from Advice* (LEA), a transfer learning method to take advantage of a library of advisors efficiently. LEA can be instantiated either as a standard *actor-based LEA-P* (for LEA-Policy), or, and this is the contribution of this paper, as a *critic-driven LEA-V* (for LEA-Value) guidance where the knowledge from advice is distilled through the proxy of the critic (value function) to enhance the student policy. This is achieved through the construction of a *guiding policy*, which reuses the best advice proposed by the library or the student policy. The rationale to use the critic (i.e., the value function of the guiding policy) as the support for guidance is that value functions are an effective medium for incorporating off-policy advice, in the context of off-policy Reinforcement Learning algorithms.

The paper is organized in the following way : The context is introduced in Section 2. The rationale for LEA, as well as the complete algorithm, including its two variants (actor-based and critic-based guidance) are thoroughly introduced in Section 3. Section 4 presents the experimental validation of LEA on standard control benchmarks (see Fig 6), in particular demonstrating that actor-based approaches struggle to capture information from sub-optimal advice. In contrast, the critic-based version of LEA provides positive transfer even in the face of unreliable advice. Section 5 discusses these results in the light of closely related works, and Section 6 concludes the paper, sketching directions for future work.

2 Background

2.1 Reinforcement Learning

Reinforcement Learning [SB18] is a framework designed to learn the behavior of an agent by trial and error through interaction with an environment. This interaction is described by a Markov Decision Process, i.e., a 5-tuple $\big(\mathcal{S}, \mathcal{A}, T, R, \gamma\big)$. When the agent is in state $s \in \mathcal{S}$, following a policy $\mu : \mathcal{S} \to \mathcal{A}$, it performs action $\mu(s)$, alters the environment, going into a new state determined by the transition distribution $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$. It then receives an instantaneous reward r determined by the function $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$. $\gamma \in [0, 1]$ is the so-called *discount factor*.

Reinforcement Learning learns a policy μ by maximizing a performance J (or minimizing a loss $\mathcal{L}$), commonly set as the discounted sum of rewards. The action-value function $Q^\mu(s, a) = \mathbb{E}\left(\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s, a_t = a\right)$ evaluates the expected returns in the context s if following policy μ after taking action a. The value [1] function Q^μ is an efficient way to design a policy by taking the action which maximizes the value and leads to two dominant learning paradigm in Reinforcement Learning known as Policy Iteration and Value Based learning scheme.

In the context of Transfer Learning, a policy may exploit information from previously solved source tasks, possibly different from the current target task. As a result, it is useful to define a task as an MDP. This representation helps us to understand the relationship between source and target. In this work, we restrict source tasks (related to advisors) to share the same state and action spaces and only differ from the target task in terms of reward R_i or transition T_i. This choice justifies the ability of advisors to act in a target environment, thus to give advice (without assumption on the quality of the advice given). Note that the relative freedom about source tasks comes at the price of potentially unreliable advisors despite near-optimal behavior in their corresponding tasks.

2.2 Policy Iteration and Value Based Algorithms

Among Reinforcement Learning approaches to learn an adequate behavior, value-based methods and actor-critic are known to be particularly efficient. Value-Based Approaches [WD92, MKSR15, FPT15] learn an optimal value function Q^* by interaction with the environment. Given the optimal value, it is relatively easy to derive the associated optimal policy, for example in the case of Q-Learning [WD92] approaches : $\mu^* = \arg\max_\mu Q^\mu(s, a), \forall s \in \mathcal{S}, a \in \mathcal{A}$. Q-learning is a common approach to learn the optimal value function only by interaction with the environment, leveraging the recursive form of the value function :

$$Q_*(s, a) = R(s, a) + \gamma \sum_{s' \in \mathcal{S}} T(s, a, s') \max_{a' \in \mathcal{A}} Q_*(s', a')$$

to learn to minimize the so-called Bellman residual :

$$\mathcal{L}(\theta) = \mathop{\mathbb{E}}_{(s,a,r,s') \sim U(D)} \Big[\big(r + \gamma \max_{a'} Q(s', a'; \theta^-) - Q(s, a; \theta) \big)^2 \Big]$$

1. *Notation detail :* Value function refers to Q in this work.

(a) Finger (b) Reacher (c) Hopper (d) Walker2d (e) HalfCheetah

FIGURE 1 – **Benchmark Environments.** List of environments used to evaluate our method. We apply our transfer learning approach LEA to several tasks based on the displayed environment.

Despite Q-Learning extensions exist to continuous action space [VWMM20, RCA+19], computation of the Bellman residual is usually intractable. In such context where the evaluation of the value of each potential action is difficult, Policy Iteration schemes [SB18] are commonly used.

Policy Iteration gradually improves the policy by repeatedly applying two steps : the *evaluation step* evaluates the current policy, and the *policy improvement step* modifies the policy (e.g., following the gradient of the loss). In continuous state space, the loss $\mathcal{L}_\mu$ is defined by : $\mathcal{L}_\mu(\theta) = -\int_{\mathcal{S}} \rho^\mu(s) Q^\mu(s, \mu_\theta(s)) ds$ where θ, μ respectively refer to the policy parameterization and a deterministic policy, and ρ^μ is the discounted state distribution induced by the policy μ. Because computing such expectation is untractable, we will use the proxy proposed by DDPG [LHP15], that uses a *replay buffer* $\mathcal{D}$ to compute an approximation thereof :

$$\mathcal{L}_\mu(\theta) = \mathop{\mathbb{E}}_{s\sim D} \left[-Q^\mu \left(s, \mu_\theta(s) \right) \right] \tag{1}$$

Whatever powerful they are, the above baseline strategies for Reinforcement Learning do not have any built-in mechanism to incorporate advice and bias the learning process of the agent toward the solution - that we will introduce now.

Algorithm 1 LEA (Library $\{\mu_i, i \in [1, n]\}$, $\epsilon_{\text{advisor}}$): returns μ_0, learned policy.

1: **Init**: $\mu_0 \leftarrow$ random policy; $s \leftarrow s_0$; $Q \leftarrow 0$; $\mathcal{D} \leftarrow \emptyset$
2: **for** $t \leftarrow 0; t < T; t \leftarrow t + 1$ **do** ▷ run for T time steps
3: (LEA-V) $\begin{cases} i^* = \arg\max_{i\in[0,n]}\{Q(s, \mu_i(s))\}) & \text{with prob. } 1 - \epsilon_{\text{advisor}} \\ i^* \sim \mathcal{U}\{1, n\} & \text{with prob. } \epsilon_{\text{advisor}} \end{cases}$
3: (LEA-P) $i^* = 0$ ▷ Choose the student policy (index $i^* = 0$)
4: $(s_{t+1}, r_{t+1}) \leftarrow Env(s, \mu_{i^*}(s))$ ▷ do chosen action, get actual reward
5: $\mathcal{D} \leftarrow \mathcal{D} \bigcup \{(s, \mu_{i^*}(s), r_{t+1}, s_{t+1})$ ▷ store in replay memory
6: **for** all (s_k, a_k, r, s^+) in mini-batch $\mathcal{B}$ from $\mathcal{D}$ **do** ▷ replays
7: $a^+ \leftarrow \mu_0(s^+)$
8: $\overline{Q}(s_k, a_k) \leftarrow r + \gamma Q(s^+, a^+)$
9: $\mathcal{L}_Q \leftarrow \frac{1}{2|\mathcal{B}|} \sum_{\mathcal{B}} ||Q(s_k, a_k) - \overline{Q}(s_k, a_k)||_2^2$
10: $\mathcal{L}_\mu \leftarrow -\frac{1}{|\mathcal{B}|} \sum_{\mathcal{B}} Q(s_k, \mu_0(s_k))$
11: (LEA-P) $\mathcal{L}_\mu \leftarrow \mathcal{L}_\mu + \frac{\beta_A}{|\mathcal{B}|} \sum_{\mathcal{B}} \|\mu_0(s_k) - \mu_{i^*}(s_k)\|_2^2$ ▷ see line 3

FIGURE 2 – LEA **Pseudo-code, with two variants**, LEA-P and LEA-V. At the end of each time step, $\mathcal{L}_Q$ and $\mathcal{L}_\mu$ are used to update Q and μ respectively (not detailed here). In the case we commit to an advisor during C timesteps, Line 3 and 4 are simply ignored and the agent follows the advisor it commits to.

3 Learning from Advice

We postulate that although advisor policies may have been trained on source tasks highly dissimilar to the target task, their advice may still be valuable. First, in the early steps, they provide efficient guidance by being merely more relevant than an almost random policy, especially if some advisors are relevant for the target task. Secondly, and more importantly, they can be used to efficiently bias the exploration of a learning agent.

We propose, in this work, to aggregate both the student policy and the advisor policies into a single *guiding policy*, which incorporates the knowledge of both the student policy and its advisors. This aggregation is only used during training, in order to smoothly guide the student policy to surpass the library of advice. Notably, we will demonstrate that when the current value function is used as a proxy for evaluating possible guidance, positive transfer is most likely to take place : this is the crux of our contribution.We hypothesize that the critic-based guidance induces smooth improvements without being limited by the sub-optimality of source policies. Furthermore, and opposite to most other Transfer Learning algorithms [FV06a, SATK18], our approach does not assume the availability of advisors at test time.

After introducing the general principles of Actor- and Critic-based guidance, we will present both variants of LEA, and argue that the latter (LEA-V) should be more robust for guidance in front of unreliable advice.

3.1 Problem Definition

Assuming prior experience in the form of a library of policies $\mathcal{E} = (\mu_i)_{i\in[1,N]}$, our objective is to reuse such knowledge to guide a new policy for a different task. We postulate (even if it is not required) that the library is available at each time-step and restrict advice, given an observation, to be in the form of actions. The main issue we need to tackle here lies in the ability to leverage advice (from potentially sub-optimal advisors) with no prior information about the context used to train the advisors. Hence, even if the advisors have been trained on profoundly different tasks, we do not want to use any prior knowledge about similarities between tasks (which is difficult to evaluate and, simply not available in the general case).
Finally, in the absence of prior information (e.g., on tasks proximity for instance), negative transfer can indeed occur in standard Transfer Learning algorithms. The identification and reuse of valuable advice is hence crucial to ensure positive transfer.

3.2 Rationale

How to evaluate the relevance of advice given off-policy [2]. Ideally, we should score advice based on the improvement they can deliver to the agent. In other words, assessing the value of an advisor is equivalent to predicting its future return if it were followed. This prediction is intricate because it relies on the expected return of the policy in a state s if the advice b were followed (which is the value function of the student policy). If this policy is inadequate, the value of the advisor's advice cannot be assessed reliably.

In this work, we study two simple approaches to incorporate advisors to enable efficient evaluation of their advice. First, we propose to imitate the best advice (according to its empirical value), we refer to such approach as actor-based guidance. Secondly, we reuse the advice (without changing the student policy) in the form of Policy Reuse [FV06a, FV06b, RHR16, LGZZ19] and guide the student through the use of the critic; we refer to such approach as critic-based guidance. A core difference with the traditional Policy Reuse approach in our work lies in the fact that the student is directly trained to surpass advisors through maximization of the value function.

3.3 Actor Based Guidance

Guidance by imitation of the most promising advice [NMA18] is a simple approach to transfer knowledge from a library of advisors. The resulting loss can be used to guide the student policy :

$$\mathcal{L}_{IL}(s,\theta) = \beta_A \|\mu_\theta(s) - \underset{a\in\mathcal{E}(s)\cup\{\mu_0(s)\}}{\arg\max} Q^\mu(s,a)\|_2^2 \quad (2)$$

Such guidance is commonly incorporated as a regularization term with the traditional reinforcement learning objective : $\mathcal{L} = \mathcal{L}_{RL} + \mathcal{L}_{IL}$.

As usual, the regularization hyperparameter β_A is difficult to tune because it is responsible for the strength of the imitation. Furthermore, such actor-driven guidance may be poorly suited to tackle unreliable advice. An alternative is to learn from the critic to alleviate such issues.

3.4 Critic Based Guidance

The objective is to indirectly guide the agent via the maximization of the value function over the guiding policy $\tilde{\mu}$. In contrast to the actor-based approach, critic-based approaches do not seek to imitate the most promising advice but rather to directly outperform the library of advisors (including the current student policy).

We propose a general recipe for critic-based guidance, given a library of advisor policies, within the actor-critic framework. The idea is to iterate over policy evaluation (over the *guiding policy* $\tilde{\mu}$) and policy improvement over the student policy. Policy Evaluation is based on the minimization of the mean squared Bellman residual commonly used in TD learning methods [Sut] :

$$\epsilon^2 = \underset{\substack{s\sim\rho^{\tilde{\mu}}\\ a\sim\tilde{\mu}}}{\mathbb{E}} \left[\left(Q^\phi(s,a) - Q^{\tilde{\mu}}(s,a)\right)^2\right] \quad (3)$$

where ϕ denotes the parameterization of the function approximating the true value $Q^{\tilde{\mu}}$. Policy Improvement is based on the improvement of the student policy based on the maximization of the value function over the library :

$$\mu_{k+1}(s) = \arg\max_a Q^{\tilde{\mu}}(s,a) \quad \forall s \in \mathcal{S} \quad (4)$$

Equation (4) constraints the student policy (μ) to perform better than the advice policies. The advantage is to leverage the value function as a guidance support, which carries more information than imitation learning. This learning scheme, although applicable in the tabular case, does not scale to continuous state and action spaces.

3.5 The LEA Algorithms

In previous section, we introduced the generic core of learning-by-advice algorithms and provided a learning scheme working in tabular environments. In this section, we will present two practical implementations in the context of

2. Understand an action that may come from a source policy highly dissimilar to the on-policy one (the student policy).

complex, continuous state-action spaces – the LEA algorithms, with two variants, the Actor-based LEA-P and the Critic-based LEA-V [3].

The pseudo code of LEA is given in figure 2 : the algorithm learns the student policy termed μ_0, initialized to a random policy in Line 1, where the initial state is set to s_0, the Q-function to 0, and the replay buffer $\mathcal{D}$ is emptied. The main loop (line 2-11) starts in state s and receives the corresponding advice $\mu_i(s)$. The first question is to determine which advice to follow (line 3).

3.6 Evaluation of the Advice Value

LEA-V guides a student policy via the use of an aggregation of the advisors and the student policy. The aggregation is a function that outputs the probabilities of selecting any given advice (including the action of the student). As a result, the aggregation should enable the reuse of the most promising advisor and ensure adequate evaluation of the value of advice.

Ideally, the value of advice should be assessed by the optimal value function Q^*, and the best action among a set of advice $A_d = \{a_1, ..., a_N\}$ of possible actions should be estimated as : $\arg\max_{a \in A_d} Q^*(s, a)$. Unfortunately, both the exhaustive evaluation and the knowledge of the optimal value function is unrealistic.

The value of advice could be defined as the future discounted return, following this advice. However, this evaluation might be highly biased, resulting in the inability to choose the best advice correctly. If the current policy is weak, the advice given the policy may be weak independently of its inner value (concerning the optimal policy or simply the value under the advisor's policy) because evaluated concerning the value function of the current policy.

The core problem is the evaluation of off-policy advice, assuming interacting with our student policy. We solve this problem partially by actively reusing advisor policies (by the proxy of the guiding policy).

Indeed, suppose that the action recommended by the student policy μ_0 is a_0. Assuming the student implements an actor-critic algorithm, we have access to an approximation of the policy and the value function of the student. The improvement of action a_i w.r.t. a_0 can be estimated by $Q(s, a_i) - Q(s, a_0)$. If it is positive, we can consider a_i as a promising action worth to be reused. Hence the most promising action is simply the action which provides the most substantial improvement, namely : $a^* = \arg\max_{a_i \in A_d} Q(s, a_i)$ But this action should only be used if better than a_0, the one given by the current student policy μ_0, i.e., if $Q(s, a^*) > Q(s, a_0)$. This can be wrapped up by

$$a^* = \underset{a_i \in A_d \cup \{a_0\}}{\arg\max}\ Q(s, a_i) \tag{5}$$

which is implemented in Lines 3 and 5 of Algorithm 1, Line 5 performing one step using this best action.

Note that 5 can be viewed as defining the *guiding policy* $\tilde{\mu}$ resulting in the following learning dynamics : When the student is poorly relevant, reusing advice is probably more useful. As the student learns, it will eventually surpass the advisors (through critic updates), and asymptotically, $\tilde{\mu}$ will converge to μ, and advisors will be ignored.

In order to avoid to weaken the exploration of Reinforcement Learning algorithm, a random policy is added to the library. Furthermore, in practice, we add a slight chance to act according to the random policy similarly to an epsilon-greedy strategy of Deep Q Learning approaches [MKSR15].

3.7 Policy Iteration with Advice

In order to apply LEA to challenging environments, we need to adapt the policy evaluation and policy improvement stages slightly.

The policy evaluation stage updates the value function by minimizing the mean squared Bellman residual error see in equation (3). We will use a standard Deep Reinforcement Learning scheme by using the so-called experience replay [Lin92] : a replay memory $\mathcal{D}$ is maintained, and stores all actual transitions in the form of tuple (s, a, r, s^+) (line 5). This replay memory is used Lines 6-8 : a mini-batch of transitions is randomly drawn from $\mathcal{D}$, and the Bellman residual is computed using the following equation, instead of the intractable 3.

$$\epsilon^2 = \mathbb{E}_{(s,a,r,s^+) \sim \mathcal{D}} \left[\left(Q(s, a) - r - \gamma Q(s^+, a^+) \right)^2 \right] \tag{6}$$

These a^+ are then used to compute an approximation of the mean squared Bellman residual (Line 9, using the results of Line 8 over $\mathcal{B}$).

The policy improvement specified by equation (4) is intractable for continuous state and action spaces. Previous approaches like DDPG [LHP15] prescribe to update the student in the direction of the gradient of the value function. We reuse the policy improvement specified in DDPG (Line 10), as is for the Critic-based LEA-V, adding the regularisation term given by 2 (Line 11, LEA-P only) in the case of our Actor-based variant LEA-P.

3. While LEA-V is the main contribution of this paper as a critic-based guidance method, LEA-P is used as a baseline for actor-based guidance, so both will be compared in the experimental study.

3.8 Enhancing Advice Evaluation via Commitment.

Since our algorithm deals with the evaluation of advice, we add a practical term called *commitment*. The commitment value is a term that specifies the number of time-steps an advisor is followed. Shortly, the commitment specifies a degree of adherence to an advisor's actions and it is used to improve the evaluation of advisors, thus improving the overall quality of our algorithm by interpolating between full reuse of an advisor and one-step advise reuse. In our work, the commitment is designed to improve the evaluation of advice via the value of the guiding policy Q^{μ} and incidentally to improve exploration bias. In practice, we found a commitment value of 5 to be a conservative setting to improve our method LEA-V.

4 Experiments

LEA algorithms rely on the identification of the best advice and the best use of valuable information from them. The objectives of the experimental study hence are i) to evaluate if LEA is robust to sub-optimal advice, and ii) to evaluate if LEA can provide valuable guidance when proposed unreliable advice.

4.1 Experimental Settings.

Our experimental study uses two Robotics Benchmark suites based on the physics engine Mujoco [TET12], namely the DeepMind Control Suite [TD$^+$18], and a Multi-Tasks Mujoco suite [HC$^+$17]. Both suites provide challenging environments with continuous state-action spaces. They are illustrated on figure 1.
We will compare the Critic-based LEA-V with its Actor-based counterpart LEA-P, and with the following residual policy learning approaches : RPL [SATK18] when only one advisor is available and A2T [RPRK15] otherwise (described in more details in section 5). Because LEA is based on the DDPG algorithm, we will also validate it by demonstrating the improvement over the bare application of DDPG (not using any advice). Furthermore, when appropriate, we will compare all the above algorithm to a very naive transfer learning approach, termed *fine-tuning*, which runs DDPG after initializing the policy with that of the advisor [4].

All advisors have been trained with TD3 [FHM18] to provide adequate expertise on the source tasks. To make the evaluation fair, all the above approaches mimic the DDPG algorithm : same neural network architecture, and same hyperparameters [5]. Residual approaches, RPL and A2T, learn their residual policies upon one or multiple base policies using DDPG. Our approach reuses the training of DDPG with slight changes to incorporate advice in the training loop.

All figures are better seen in color. Unless otherwise specified, all plots represent the mean normalized performance over 5 independent runs, smoothed with an exponential moving average (on 10 epochs). The error bars are not represented on the figures for readability despite limited space. However, on this limited number of runs, the variance of LEA was much smaller than that of A2T, and similar to that of DDPG.

4.2 Sensitivity to unreliable advice

Leveraging information from prior experience is the principal subject of Transfer Learning. In the experiments, experts are trained in different contexts, resulting in possibly unreliable advice. Robustness to unreliable advice becomes a crucial property to avoid a negative transfer. We study the ability to ignore irrelevant advice and compare the ability for transfer-learning to reuse advisors (with different levels of relevance to the task). In this setting, source and target environment are the same and the focus is put on the advisor relevance.

4.2.1 Regularization Strength in Actor Based Guidance

The regularization parameter (β_A in 2) characterizes LEA-P. We test several regularization strengths on an imitation learning task where the learning agent receives advice from a sub-optimal expert.

In 3, one can distinguish two learning regimes. The values $\beta_A \in [0, 10]$ achieve the best asymptotic performance. At the same time, these runs provide no improvement in terms of learning dynamics compared to $\beta_A = 0$ (no imitation). On the opposite, with $\beta_A \geq 100$, the agent starts at a higher level of performance (jumpstart performance) but reaches a sub-optimal threshold around a score of 430, close to the expert's performance.

If β_A is too small, the agent is unable to extract valuable information from the advice, and the training is similar to training without supervision. On the contrary, if too large, the agent bluntly imitates the experts. This observation promotes the idea that the Actor Based method LEA-P is challenging to tune in order to extract valuable information from a sub-optimal expert. In the remaining of this paper, β_A

4. This fine-tuning approach, contrary to residual and LEA approaches, assumes complete knowledge of the advisor network, here the same neural network structure.

5. Note that A2T needs an additional network for aggregating the advisors.

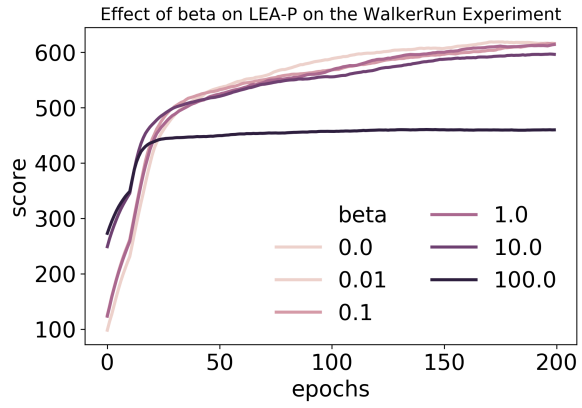

FIGURE 3 – Sensitivity of LEA-P w.r.t. β_A on FingerTurn (hard version). The darker the color, the higher the value for β_A.

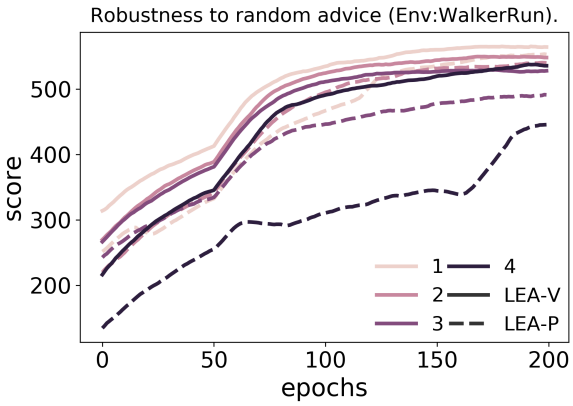

FIGURE 4 – Robustness to random advice on WalkerRun (random advice do not make sense for Residual Learning).

will be set to 1, a good trade-off between asymptotic performance and actor-based regularization, ensuring some degree of robustness. We will now compare our approach with experts of various levels of relevance to the task at hand (from weak to strong).

4.2.2 Learning with Random Advice

We first evaluate the robustness of LEA-V and LEA-P in one of the worst scenarios, in which the advice are random. The goal is to assess the ability to ignore poor advice. First, figure 4 shows that both LEA-P and LEA-V algorithms are sensitive to the number of random advice given : Both approaches obtain their best results with one (random) advisor, and their performances decrease with the number of advisors. However, LEA-V achieves asymptotically near-optimal performance for all settings. Furthermore, the decrease of performance with the number of random advice is more important for LEA-P, highlighting one limitation of the imitation based approach.

If the evaluation of advice is not accurate, increasing the number of advice will not help, and might even end up increasing the possibility of following poor advice. If such a toy context is rarely encountered in practice, guiding policy with sub-optimal advice is a requirement to leverage information without the need for prior information about advisors, barely reliably available : it is challenging to evaluate the relevance of an advisor to a task different from the one it was trained for. The next experiment will reinforce this conclusion by comparing the transfer provided by various levels of expertise of the advisors.

4.2.3 Sensitivity w.r.t. Advice Expertise

In this setting, we provide the learning agent with advice from a weak, a medium, and a strong advisor : all are trained DDPG, in the same environment, but during 1, 50 or 200 episodes. Note that the weak advisor is not random anymore, though not achieving any good result yet. 5 shows how the algorithms react in the presence of these differently skilled advisors.

Unsurprisingly, better advice leads to better results for all algorithms. In terms of learning dynamics, when the advice is coming from a competent expert, policy-driven approaches (LEA-P and RPL) present a slightly faster time-to-threshold, but no significant asymptotic improvement. On the contrary, with weak advice, both learning dynamics and asymptotic performance degrade. Bad advice slows the time to threshold and leads to sub-optimal asymptotic performance. However, LEA-V demonstrates its robustness to poor advice, whereas the Residual approach is not able to learn an adequate corrective policy. To shade previous conclusions, if some actor-based guidance is used with a moderately expert advisor, both algorithms present the same learning dynamics.

Such result suggests that both residual and policy-driven guidance have difficulty not to imitate the advisor (independently of its relevancy). Thus when the advisor is unreliable, the result will inevitably be worse than those of the critic-based approach LEA-V. The fundamental reason is that imitation The next experiment will remove this limitation, and evaluate the performance of both approaches in a more challenging context, where experts are trained on different tasks, though in the same setting.

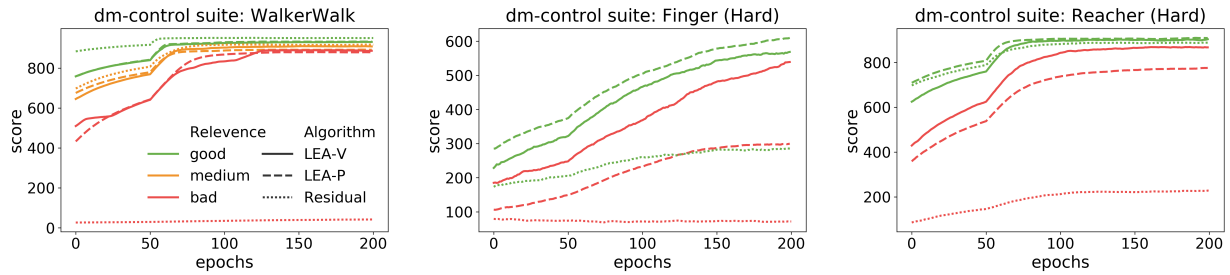

FIGURE 5 – **Transfer Learning with Optimal and Sub-optimal Advice**. Comparative results when using optimal and sub-optimal advisors in different environments for LEAs and Residual Learning.

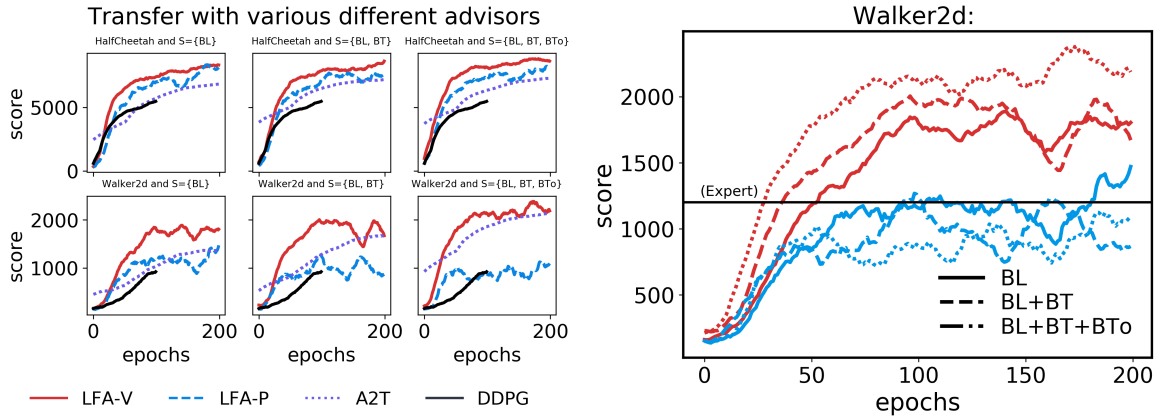

FIGURE 6 – **Transfer Learning with several advisors.** Figure on the left : Transfer Learning with gradually more advisor provided (increasing from left to right). Figure on the right : Aggregation of the Walker2d results with LEA-V and LEA-P with three settings : one, two and three advisors.

4.3 Sensitivity to library length

Few experts are valuable in all contexts. In order to tackle a complex environment, we should reuse any available information. We compare to what margin the number of advice provided can impact the learning process of our algorithm and actor-based guidance. Results of the experiment are displayed in figure 6.

LEA-V benefits from the use of multiple advisors in terms of time-to-threshold. The sampling efficiency increases with the number of advisors. The asymptotic performance seems not to be affected (or only by a limited margin) with the addition of the last advisor. On the contrary, LEA-P is limited by the performance of the advisor (even with our conservative value of β as chosen in figure 4.2.1).

The experiment confirms that the number of advisors does not limit our approach. More interestingly, asymptotic performance is not changed with the addition of an advisor, the sample efficiency increases. One possible reason is that even if the advisors are sub-optimal, during the beginning of the training, they provide valuable advice. With a larger pool of advisors, the same phenomenon arises at the exception that the best advice among sub-optimal ones will be selected, increasing the sample efficiency.

In summary, the results displayed on the right of figure 6 show the core difference between LEA-V and LEA-P. They validate our assumption that the advisors limit LEA-P whereas LEA-V provides, in addition to the leveraging of multiple advisors, substantial gain in terms of asymptotic performance.

5 Related Work

Policy Reuse [FV06a, FV06b, FV05] solves the target task by reusing actions proposed by a pool of source policies in order to bias exploration. Policy Reuse uses a practical learning bias to reuse actions efficiently. Because similar tasks should have similar optimal policies, reusing ac-

tions from the policy trained on the closest tasks should be efficient. Unfortunately, the similarity between tasks is difficult to assess. As a result, learning to select the most valuable policy for reuse is required. Value functions are an adequate filter to select valuable advice (without explicitly defining an intra-task similarity) and have been used in the context of the reuse of options [LGZZ19] and actions [RHR16, KM$^+$19]. A core difference with LEA-V is the reuse of the critic to directly improve the student policy, which is then able to surpass the advisors, whereas Policy Reuse provides indirect guidance by partially alleviating the exploration-exploitation trade-off.

Residual Learning [SATK18,JB$^+$19] proposes to guide an agent by learning a corrective policy (the *residual*) to improve its overall performance on a novel but similar tasks. Residual approaches commonly work with an already well-performing base policy. If the base policy is weak, a negative transfer may occur due to the difficulty of finding an adequate corrective policy. Furthermore, to alleviate the restriction of using only one source policy, several works combine aggregation steps (which combine the pool of advice received into a single action) with a corrective one [RPRK15, BY$^+$19] which induce additional learning costs. LEA does not rely on any additional learned aggregation layer but corrects the base policy through the proxy of the value function to improve the student policy.

Learning From Demonstration leverages expert's demonstration by trying to reduce the gap between the experts and the reinforcement learning agent. [HVP$^+$17] and [VHSW17] show the ability of imitation learning to handle complex tasks. [NMA18] tackles situations where an expert is sub-optimal and proposes a filtering approach to filter weak demonstration. In constrast, our method directly use policy as the support for the transfer. This choice allows to easily the reuse of advice theis evaluatation within the context of the current policy.

Distillation and Knowledge Transfer [SHZ$^+$18, CJJ$^+$18] study the transfer from source to task policy in the context of multi-task learning. These works assume that the source policy contains valuable information to learn the target policy. In contrast, LEA-V provides a more direct way to leverage a pool of policies. LEA does not assume that experts are valuable and uses the critic to guide the student policy without additional constraint.

6 Conclusion

This work addressed the problem of learning from possibly unreliable advisors, trained in potentially different contexts than the current one. We demonstrated that directly modifying the policy is not robust to poor advice. We proposed LEA, an alternative approach based on a guiding policy using the current critic applied to the advisors. We demonstrated that improving the student policy thanks to the value function of the guiding policy generally leads to positive transfer, thus outperforming the state-of-the-art baselines. Furthermore, LEA-V, the critic-guided variant of LEA, is competitive with previous residual approaches, without the constraint of the availability of the advisors at test time.

Ultimately, the main issue with LEA-like approaches remains how to characterize and estimate the value of the advice which will be study in futher work.

Références

[ABC$^+$18] Marcin Andrychowicz, Bowen Baker, Chociej, et al. Learning dexterous in-hand manipulation. *arXiv e-prints*, page arXiv :1808.00177, 2018.

[BY$^+$19] Mohammadamin Barekatain, Ryo Yonetani, et al. Multipolar : Multi-source policy aggregation for transfer reinforcement learning between diverse environmental dynamics. *ArXiv :1909.13111*, 2019.

[CJJ$^+$18] Wojciech Marian Czarnecki, Siddhant M. Jayakumar, Max Jaderberg, et al. Mix&match - agent curricula for reinforcement learning. *CoRR*, abs/1806.01780, 2018.

[FHM18] Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning*, pages 1582–1591, 2018.

[FPT15] Roy Fox, Ari Pakman, and Naftali Tishby. G-learning : Taming the noise in reinforcement learning via soft updates. *CoRR*, abs/1512.08562, 2015.

[FV05] Fernando Fernández and Manuela Veloso. Building a library of policies through policy reuse. Technical report, CMU-CS-05-174, Carnegie Mellon University, 2005.

[FV06a] Fernando Fernández and Manuela Veloso. Policy reuse for transfer learning across tasks with different state and action spaces. In *ICML Wkp on Structural Knowledge Transfer for ML*, 2006.

[FV06b] Fernando Fernández and Manuela Veloso. Probabilistic policy reuse in a reinforcement learning agent. In *Proc. 5th AAMAS*, pages 720–727, 2006.

[HC+17] P. Henderson, W.-D. Chang, et al. Benchmark environments for multitask learning in continuous domains. *ICML Wkp on Lifelong Learning, an RL Approach*, 2017.

[HVP+17] Todd Hester, Matej Vecerik, Olivier Pietquin, Marc Lanctot, and et al. Schaul. Deep Q-learning from Demonstrations. *arXiv e-prints*, page arXiv :1704.03732, 2017.

[JB+19] Tobias Johannink, Shikhar Bahl, et al. Residual reinforcement learning for robot control. In *Proc. ICRA*, pages 6023–6029, 2019.

[K+03] Sham Machandranath Kakade et al. *On the sample complexity of reinforcement learning*. PhD thesis, University of London London, England, 2003.

[KM+19] Andrey Kurenkov, Ajay Mandlekar, et al. Acteach : A bayesian actor-critic method for policy learning with an ensemble of suboptimal teachers. *ArXiv :1909.04121*, 2019.

[LGZZ19] Siyuan Li, Fangda Gu, Guangxiang Zhu, and Chongjie Zhang. Context-aware policy reuse. In *Proc, 18th AAMAS*, pages 989–997, 2019.

[LHP15] Timothy P Lillicrap, Jonathan J Hunt, and et al Pritzel. Continuous control with deep reinforcement learning. *arXiv preprint arXiv :1509.02971*, 2015.

[Lin92] Long-Ji Lin. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine learning*, 8(3-4) :293–321, 1992.

[MK13] Volodymyr Mnih and et al. Kavukcuoglu. Playing atari with deep reinforcement learning. *arXiv e-prints*, page arXiv :1312.5602, 2013.

[MKSR15] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, and et al. Rusu. Human-level control through deep reinforcement learning. *Nature*, 518(7540) :529–533, 2015.

[NHR99] Andrew Y. Ng, Daishi Harada, and Stuart J. Russell. Policy invariance under reward transformations : Theory and application to reward shaping. In *Proc. 16th ICML*, pages 278–287, 1999.

[NMA18] Ashvin Nair, Bob McGrew, and et al Andrychowicz. Overcoming exploration in reinforcement learning with demonstrations. In *2018 ICRA*, pages 6292–6299, 2018.

[RCA+19] Moonkyung Ryu, Yinlam Chow, Ross Anderson, Christian Tjand raatmadja, and Craig Boutilier. Caql : Continuous action q-learning. *arXiv e-prints*, page arXiv :1909.12397, 2019.

[RHR16] Benjamin Rosman, Majd Hawasly, and Subramanian Ramamoorthy. Bayesian policy reuse. *Machine Learning*, 104(1) :99–127, 2016.

[RPRK15] Janarthanan Rajendran, P. Prasanna, Balaraman Ravindran, and Mitesh M. Khapra. ADAAPT : A deep architecture for adaptive policy transfer from multiple sources. *CoRR*, abs/1510.02879, 2015.

[SATK18] Tom Silver, Kelsey R. Allen, Josh Tenenbaum, and Leslie Pack Kaelbling. Residual policy learning. *CoRR*, abs/1812.06298, 2018.

[SB18] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning : An Introduction*. MIT Press, 2018.

[SHZ+18] Simon Schmitt, Jonathan J. Hudson, Augustin Zídek, , et al. Kickstarting deep reinforcement learning. *CoRR*, abs/1803.03835, 2018.

[Sut] Richard S Sutton. Learning to predict by the methods of temporal differences. *Machine learning*.

[TD+18] Yuval Tassa, Yotam Doron, et al. Deepmind control suite. *CoRR*, abs/1801.00690, 2018.

[TET12] E. Todorov, T. Erez, and Y. Tassa. Mujoco : A physics engine for model-based control. In *PRoc. IROS*, pages 5026–5033, 2012.

[VHSea17] Mel Vecerik, Todd Hester, Jonathan Scholz, and et al. Leveraging Demonstrations for Deep Reinforcement Learning on Robotics Problems with Sparse Rewards. *arXiv e-prints*, 2017.

[VHSW17] Matej Večerík, Todd Hester, Jonathan Scholz, and et al. Wang. Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards. *arXiv preprint arXiv :1707.08817*, 2017.

[VWMM20] Tom Van de Wiele, David Warde-Farley, Andriy Mnih, and Volodymyr Mnih. Q-learning in enormous action spaces via amortized approximate maximization. *arXiv e-prints*, page arXiv :2001.08116, 2020.

[WD92] Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3-4) :279–292, 1992.